



# SPYWOLF

## Security Audit Report



Audit prepared for  
**Chain Factory**

Completed on  
**December 31, 2023**

@SPYWOLFNETWORK



@SPYWOLFNETWORK



SPYWOLF.CO





# OVERVIEW

This goal of this report is to review the main aspects of the project to help investors make an informative decision during their research process.

You will find a a summarized review of the following key points:

- ✓ Contract's source code
- ✓ Owners' wallets
- ✓ Tokenomics
- ✓ Team transparency and goals
- ✓ Website's age, code, security and UX
- ✓ Whitepaper and roadmap
- ✓ Social media & online presence

“

*The results of this audit are purely based on the team's evaluation and does not guarantee nor reflect the projects outcome and goal*

- SPYWOLF Team -

”





# TABLE OF CONTENTS

---

Project Description	01
Contract 1	02
Info & Stats	02-04
Found Threats	05-08
Contract 2	09
Info & Stats	09-11
Found Threats	12-13
Contract 3	14
Info & Stats	15-16
Found Threats	17-18
Tokenomics	19
Website Analysis	20
Social Media & Online Presence	21
About SPYWOLF	22
Disclaimer	23



# CHAIN FACTORY



## PROJECT DESCRIPTION

"With ChainFactory, users can choose from a variety of customizable templates and features, making it simple to create contracts tailored to your specific needs. It is designed to be user-friendly and intuitive, guiding users through the entire process step-by-step, providing a centralized platform to create, deploy, and manage your Smart-Contracts with ease."

**Release Date:** Launching in 2024

**Category:** Ecosystem



# MAIN TOKEN CONTRACT

Token Name	Symbol
ChainFactory	\$FACTORY
Contract Address	
0x4ea3087b6dF5B02550d86D6cD5B246Dd70cb19F9	
Network	Language
Ethereum Sepolia TESTNET	Solidity
Deployment Date	Contract Type
Dec 24, 2023	Token with taxes
Total Supply	Status
75,000,000	Launched

## TAXES



\*Taxes can be changed in future



## Our Contract Review Process

The contract review process pays special attention to the following:

- ✓ Testing the smart contracts against both common and uncommon vulnerabilities
- ✓ Assessing the codebase to ensure compliance with current best practices and industry standards.
- ✓ Ensuring contract logic meets the specifications and intentions of the client.
- ✓ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- ✓ Thorough line-by-line manual review of the entire codebase by industry experts.

### Blockchain security tools used:

- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat



# TOKEN TRANSFERS STATS

Transfer Count	TESTNET
Uniq Senders	TESTNET
Uniq Receivers	TESTNET
Total Amount	TESTNET
Median Transfer Amount	TESTNET
Average Transfer Amount	TESTNET
First transfer date	TESTNET
Last transfer date	TESTNET
Days token transferred	TESTNET

# SMART CONTRACT STATS

Calls Count	TESTNET
External calls	TESTNET
Internal calls	TESTNET
Transactions count	TESTNET
Uniq Callers	TESTNET
Days contract called	TESTNET
Last transaction time	TESTNET
Created	TESTNET
Create TX	TESTNET
Creator	TESTNET



# FEATURED WALLETS

Owner address	0xBA799d418D1356ff5d225096d08951a3b45b6e4A
Marketing fee receivers	0xBA799d418D1356ff5d225096d08951a3b45b6e4A 0x6CBa0da9bF86076dC9Ce68082C57E1D0cE0FcF35
LP address	0x3D250979FF422D2299e3BD148e744d5e4a68EA9e

# TOP 3 UNLOCKED WALLETS

N/A	TESTNET
N/A	TESTNET
N/A	TESTNET



# VULNERABILITY ANALYSIS

ID	Title	
SWC-100	Function Default Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Passed
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed
SWC-107	Reentrancy	Passed
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Passed
SWC-116	Block values as a proxy for time	Passed
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed





# VULNERABILITY ANALYSIS

ID	Title	
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Griefing	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed
SWC-134	Message call with hardcoded gas amount	Passed
SWC-135	Code With No Effects	Low
SWC-136	Unencrypted Private Data On-Chain	Passed



# VULNERABILITY ANALYSIS

## SWC-135, severity: Low

```
function _setReflection(address token) internal {  
    require(token == address(0) || token == address(this) || token  
  
    if (token == address(0)) { token == address(this); }  
}
```

### Usage of equality comparison instead of assignment

This equality comparison doesn't have any effect. Did you mean to do assignment instead?



# MANUAL CODE REVIEW

---

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time.

We categorize these vulnerabilities by 4 different threat levels.

## THREAT LEVELS

### High Risk

---

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

### Medium Risk

---

Issues on this level are critical to the smart contract's performance, functionality and should be fixed before moving to a live environment.

### Low Risk

---

Issues on this level are minor details and warning that can remain unfixed.

### Informational

---

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.



# FOUND THREATS

## ⚠ High Risk

### Owner can change reflections token.

If router is set as reflection token, contract will halt on sell once it reaches the autoswap amount.

```
function setReflection(address token) external onlyOwner {
    require(!_renounced.DEXRouterV2);
    _setReflection(token);
}

function _setReflection(address token) internal {
    require(token == address(0) || token == address(this) || token == _dex.WSepoliaETH
    || IDEXFactoryV2(IDEXRouterV2(_dex.router).factory()).getPair(_dex.WSepoliaETH, token) != address(0),
    "No Pair");

    if (token == address(0)) { token == address(this); }

    _reflectionToken = IERC20(token);
}
```

- Recommendation:
  - Router contracts should not be set as reflection token.



# FOUND THREATS

## ⚠ High Risk

**Owner can enable/disable trading until the functionality is**

```
function setTradingStatus(bool status) external onlyOwner {
    require(!_renounced.DEXRouterV2);

    _tradingEnabled = status ? _timestamp() : 0;
}

function renounceDEXRouterV2() external onlyOwner {
    _renounced.DEXRouterV2 = true;

    emit RenouncedDEXRouterV2();
}

function _transfer(address from, address to, uint256 amount) internal virtual override {
    .....
    require((from != _dex.pair && to != _dex.pair) ||
    ((from == _dex.pair || to == _dex.pair) && _tradingEnabled > 0), "Trading disabled");
    .....
}
```

- Recommendation:
  - Considered as good practice is trading to be enabled once. Once enabled, trading should not be disabled again.



# FOUND THREATS

## ⚠ High Risk

**Owner can withdraw any tokens from the contract.**

When forced, `_taxBeneficiary.unclaimed` and `_amountForTaxDistribution` variables become out of sync.

`_taxBeneficiary.unclaimed` variable becomes higher than contract's balances. Contract will halt on sell if reflection token is set as the native \$FACTORY, once the auto swap amount is reached.

```
function recoverERC20(address token, address to, uint256 amount, bool force) external onlyOwner {
unchecked {
uint256 balance = IERC20(token).balanceOf(address(this));
uint256 allocated = token == address(this) ? _amountForTaxDistribution + _amountForLiquidity
: (address(_reflectionToken) == token ? _reflectionTokensForTaxDistribution : 0);

require((!force && balance - (allocated >= balance ? balance : allocated) >= amount)
|| (force && balance >= amount), "Exceeds balance");

if (force && (token == address(this) || address(_reflectionToken) == token)
&& balance - (allocated >= balance ? balance : allocated) < amount) {
require(!_distributing && !_swapping);

if (token == address(this)) {
uint256 pickFromAmountForTaxDistribution = amount >= _amountForTaxDistribution ?
_amountForTaxDistribution : _amountForTaxDistribution - amount;

_amountForTaxDistribution -= pickFromAmountForTaxDistribution;
allocated -= pickFromAmountForTaxDistribution;

if (balance - (allocated >= balance ? balance : allocated) < amount)
{ _amountForLiquidity -= amount >= _amountForLiquidity ?
_amountForLiquidity : _amountForLiquidity - amount; }
} else if (address(_reflectionToken) == token) {
_reflectionTokensForTaxDistribution -= amount >= _reflectionTokensForTaxDistribution ?
_reflectionTokensForTaxDistribution : _reflectionTokensForTaxDistribution - amount;
}
}

IERC20(token).transfer(to, amount);
}
```

Once out of sync, the variable cannot be synced again and that beneficiary slot will be unusable.

- Recommendation:
  - `_taxBeneficiary.unclaimed` and `_amountForTaxDistribution` variables should always in sync with each other or another approach for tax accounting should be considered.



# FOUND THREATS

## ⚠ High Risk

**Owner can blacklist address until blacklisting functionality is renounced. Blacklisted addresses are unable to sell tokens.**

```
function blacklist(address account, bool status) public onlyOwner {
    _blacklist(account, status);
}

function _blacklist(address account, bool status) internal {
    require(!_renounced.Blacklist);
    require(account != _owner && account != address(0) && account != address(0xdEaD));
    require(account != _dex.router && account != _dex.pair, "DEX router or pair");

    if (status) { require(!_whitelisted[account], "Whitelisted"); }

    _blacklisted[account] = status;

    emit Blacklisted(account, status);
}

function blacklist(address[] calldata accounts, bool status) external onlyOwner {
    unchecked {
        uint256 cnt = accounts.length;

        for (uint256 i; i < cnt; i++) { _blacklist(accounts[i], status); }
    }
}

function renounceBlacklist() external onlyOwner {
    _renounced.Blacklist = true;

    emit RenouncedBlacklist();
}
```

- Recommendation:
  - Blacklisting addresses should be automated (bot protection) with reasonable boundaries (blocks, seconds, etc.).





# FOUND THREATS

## ⚠ Medium Risk

**Owner can set buy/sell/transfer taxes up to 25%.**

Penalty taxes can be set up to 50%.

Owner can change taxes receiving wallets.

```
function renounceTaxable() external onlyOwner {
    _renounced.Taxable = true;

    emit RenouncedTaxable();
}

function setTaxBeneficiary(uint8 slot, address account,
uint24[3] memory percent, uint24[3] memory penalty) external onlyOwner {
    require(!_renounced.Taxable);

    _setTaxBeneficiary(slot, account, percent, penalty);
}

function _setTaxBeneficiary(uint8 slot, address account,
uint24[3] memory percent, uint24[3] memory penalty) internal {
    require(slot < 5);
    require(account != address(this) && account != address(0));

    taxBeneficiary storage _taxBeneficiary = _taxBeneficiary[slot];

    if (account == address(0xEaD) && _taxBeneficiary.exists && _taxBeneficiary.unclaimed > 0)
    { revert("Unclaimed taxes"); }

    _taxBeneficiary.account = account;
    _taxBeneficiary.percent = percent;
    _taxBeneficiary.penalty = penalty;

    unchecked {
        _totalTxTax += percent[0] - (_taxBeneficiary.exists ? _taxBeneficiary.percent[0] : 0);
        _totalBuyTax += percent[1] - (_taxBeneficiary.exists ? _taxBeneficiary.percent[1] : 0);
        _totalSellTax += percent[2] - (_taxBeneficiary.exists ? _taxBeneficiary.percent[2] : 0);
        _totalPenaltyTxTax += penalty[0] - (_taxBeneficiary.exists ? _taxBeneficiary.penalty[0] : 0);
        _totalPenaltyBuyTax += penalty[1] - (_taxBeneficiary.exists ? _taxBeneficiary.penalty[1] : 0);
        _totalPenaltySellTax += penalty[2] - (_taxBeneficiary.exists ? _taxBeneficiary.penalty[2] : 0);

        require(_totalTxTax <= 25 * denominator && _totalBuyTax <= 25 * denominator &&
        _totalSellTax <= 25 * denominator, "High Tax");
        require(_totalPenaltyTxTax <= 50 * denominator && _totalPenaltyBuyTax <= 50 * denominator &&
        _totalPenaltySellTax <= 50 * denominator, "High Penalty");
    }

    if (!_taxBeneficiary.exists) { _taxBeneficiary.exists = true; }

    emit SetTaxBeneficiary(slot, account, percent, penalty);
}
```

`_totalTxTax`, `_totalBuyTax`,  
`_totalSellTax`, `_totalPenaltyTxTax`,  
`_totalPenaltyBuyTax`,  
`_totalPenaltySellTax` state  
variables are only increasing.  
Once they reach the maximum  
values, no new tax beneficiaries  
can be added.

- Recommendation:
  - Considered as good practice is buy and sell fees combined not to exceed 25%.





# FOUND THREATS

## ⚠ Medium Risk

**Owner can set buy/sell/transfer adaptive taxes up to 50%.**

Adaptive taxes will be applied when the amount transferred is higher than the current adaptive tax threshold.

```
function setAdaptiveTax(uint8 level, uint256 threshold, uint24[3] memory multiplier) external onlyOwner {
    require(!_renounced.Taxable);
    require(level < 5);

    _setAdaptiveTax(level, threshold, multiplier);
}

function _setAdaptiveTax(uint8 level, uint256 threshold, uint24[3] memory multiplier) internal {
    require(multiplier[0] >= denominator && multiplier[1] >= denominator && multiplier[2] >= denominator);

    unchecked {
        require((_totalTxTax * multiplier[0]) / denominator <= 50 * denominator &&
            (_totalBuyTax * multiplier[1]) / denominator <= 50 * denominator &&
            (_totalSellTax * multiplier[2]) / denominator <= 50 * denominator, "High Multiplier");

        if (level + 1 < 5 && _adaptiveTax[level + 1].exists) { require(_adaptiveTax[level + 1].threshold > threshold); }
        if (level - 1 >= 0 && _adaptiveTax[level - 1].exists) { require(_adaptiveTax[level - 1].threshold < threshold); }
    }

    if (!_adaptiveTax[level].exists) { _adaptiveTax[level].exists = true; }

    _adaptiveTax[level].threshold = threshold;
    _adaptiveTax[level].multiplier = multiplier;

    emit SetAdaptiveTax(level, threshold, multiplier);
}
```

- Recommendation:
  - Considered as good practice is buy and sell fees combined not to exceed 25%.



# FOUND THREATS

## ⚠ Low Risk

**Owner can set max transaction amount that address can transfer as low as 0.001% of total supply, until the functionality is renounced.**

```
function renounceMaxTx() external onlyOwner {
    _renounced.MaxTx = true;

    emit RenouncedMaxTx();
}

function setMaxTxPercent(uint24 percent) external onlyOwner {
    require(!_renounced.MaxTx);

    unchecked {
        require(percent <= 100 * denominator);
    }

    _setMaxTxPercent(percent);

    emit SetMaxTxPercent(percent);
}

function _setMaxTxPercent(uint24 percent) internal {
    _maxTxPercent = percent;
    _maxTxAmount = percent > 0 ? percentage(_totalSupply, uint256(percent)) : 0;

    if (!_initialized) { emit SetMaxTxPercent(percent); }
}

function _transfer(address from, address to, uint256 amount) internal virtual override {
    .....
    require(_maxTxAmount == 0 || amount <= _maxTxAmount, "Exceeds maxTx");
    .....
}
```

- Recommendation:
  - Considered as good practice is max transaction limit to be higher than 0.1% of total supply.



# FOUND THREATS

## Informational

**Owner can set max balance (max wallet) that address can hold as low as 0.001% of total supply, until the functionality is renounced.**

```
function renounceMaxBalance() external onlyOwner {
    _renounced.MaxBalance = true;

    emit RenouncedMaxBalance();
}

function setMaxBalancePercent(uint24 percent) external onlyOwner {
    require(!_renounced.MaxBalance);

    unchecked {
        require(percent <= 100 * denominator);
    }

    _setMaxBalancePercent(percent);

    emit SetMaxBalancePercent(percent);
}

function _setMaxBalancePercent(uint24 percent) internal {
    _maxBalancePercent = percent;
    _maxBalanceAmount = percent > 0 ? _percentage(_totalSupply, uint256(percent)) : 0;

    if (!_initialized) { emit SetMaxBalancePercent(percent); }
}
```



# FOUND THREATS

## Informational

**Owner can set penalty time for sells up to 7 days from trading start timestamp.**  
If holders sell their tokens before the time set, they will be subject to penalty taxes.

```
function setEarlyPenaltyTime(uint32 time) external onlyOwner {
    require(!_renounced.Taxable);
    require(time <= 7 days);

    _setEarlyPenaltyTime(time);
}

function _setEarlyPenaltyTime(uint32 time) internal {
    _earlyPenaltyTime = time;

    emit SetEarlyPenaltyTime(time);
}
```

**Owner can withdraw ETH from the contract.**

When this function is present, in cases ETH is sent into the contract by mistake or purposefully, contract's owner can retrieve it.

```
function recoverSepoliaETH(address payable to, uint256 amount, bool force) external onlyOwner {
    unchecked {
        uint256 balance = address(this).balance;
        uint256 allocated = address(_reflectionToken) == _dex.WSepoliaETH ? _ethForTaxDistribution : 0;

        require((!force && balance - (allocated >= balance ? balance : allocated) >= amount) || (force && balance >= amount), "Exceeds balance");

        if (force && address(_reflectionToken) == _dex.WSepoliaETH && balance - (allocated >= balance ? balance : allocated) < amount) {
            require(!_distributing && !_swapping);

            _ethForTaxDistribution -= amount >= _ethForTaxDistribution ? _ethForTaxDistribution : _ethForTaxDistribution - amount;
        }
    }

    (bool success, ) = to.call{ value: amount }("");

    require(success);
}
```



# FOUND THREATS

## Informational

**Cooldown restriction will never trigger because `_holder[account].exists` will be always false.**

```
function _setCooldown(uint8 count, uint32 time, uint32 period) internal {
    require(count > 1 && time > 5);

    _cooldownTriggerCount = count;
    _cooldownTriggerTime = time;
    _cooldownPeriod = period;

    emit SetCooldown(count, time, period);
}

function _transfer(address from, address to, uint256 amount) internal virtual override {
    .....
    if (_cooldownPeriod > 0 && from != _dex.pair) {
        require(remainingCooldownTime(from) == 0, "Cooldown");

        if (_holder[from].start + _cooldownTriggerTime < _timestamp()) {
            _holder[from].count = 1;
            _holder[from].start = _timestamp();
        } else {
            if (++_holder[from].count >= _cooldownTriggerCount) { _cooldown(from); }
        }
    }
    .....
}

function _cooldown(address account) internal {
    unchecked {
        _holder[account].cooldown = _timestamp() + _cooldownPeriod;
    }
}

function remainingCooldownTime(address account) public view returns (uint32) {
    if (_cooldownPeriod == 0 || !_holder[account].exists || _holder[account].cooldown < _timestamp()) { return 0; }

    unchecked {
        return _holder[account].cooldown - _timestamp();
    }
}
```

- Recommendation:
  - For future fixes, consider restraining the cooldown variables to reasonable time. Current max value is 4294967296 seconds.

# TRANSPARENT PROXY CONTRACT

Token Name N/A	Symbol N/A
Contract Address 0x12d17EA0e9F0B2239Dcf212DA9F7a4C91f92E10e	
Network Ethereum Sepolia TESTNET	Language Solidity
Deployment Date Dec 17, 2023	Contract Type Proxy
Total Supply N/A	Status Launched

## TAXES

Buy Tax  
n/a

Sell Tax  
n/a



## Our Contract Review Process

The contract review process pays special attention to the following:

- ✓ Testing the smart contracts against both common and uncommon vulnerabilities
- ✓ Assessing the codebase to ensure compliance with current best practices and industry standards.
- ✓ Ensuring contract logic meets the specifications and intentions of the client.
- ✓ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- ✓ Thorough line-by-line manual review of the entire codebase by industry experts.

### Blockchain security tools used:

- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat





# TOKEN TRANSFERS STATS

Transfer Count	TESTNET
Uniq Senders	TESTNET
Uniq Receivers	TESTNET
Total Amount	TESTNET
Median Transfer Amount	TESTNET
Average Transfer Amount	TESTNET
First transfer date	TESTNET
Last transfer date	TESTNET
Days token transferred	TESTNET

# SMART CONTRACT STATS

Calls Count	TESTNET
External calls	TESTNET
Internal calls	TESTNET
Transactions count	TESTNET
Uniq Callers	TESTNET
Days contract called	TESTNET
Last transaction time	TESTNET
Created	TESTNET
Create TX	TESTNET
Creator	TESTNET



# FEATURED WALLETS

Owner address	0xBA799d418D1356ff5d225096d08951a3b45b6e4A
Current proxy implementation logic address	0xF6Eb5258748A150313F8B2E304F22646f8CB3dda
LP address	N/A

# TOP 3 UNLOCKED WALLETS

N/A	N/A
N/A	N/A
N/A	N/A





# VULNERABILITY ANALYSIS

ID	Title	
SWC-100	Function Default Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Passed
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed
SWC-107	Reentrancy	Passed
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Passed
SWC-116	Block values as a proxy for time	Passed
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed



# VULNERABILITY ANALYSIS

ID	Title	
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Griefing	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed
SWC-134	Message call with hardcoded gas amount	Passed
SWC-135	Code With No Effects	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed



# FOUND THREATS

## High Risk


No high risk-level threats found in this contract.

## Medium Risk

No medium risk-level threats found in this contract.

## Low Risk

No low risk-level threats found in this contract.



# PROXY IMPLEMENTATION OF FACTORY CONTRACT

Token Name N/A	Symbol N/A
Contract Address 0xF6Eb5258748A150313F8B2E304F22646f8CB3dda	
Network Ethereum Sepolia TESTNET	Language Solidity
Deployment Date Dec 23, 2023	Contract Type Contracts factory
Total Supply N/A	Status Launched

## TAXES

Buy Tax  
n/a

Sell Tax  
n/a



## Our Contract Review Process

The contract review process pays special attention to the following:

- ✓ Testing the smart contracts against both common and uncommon vulnerabilities
- ✓ Assessing the codebase to ensure compliance with current best practices and industry standards.
- ✓ Ensuring contract logic meets the specifications and intentions of the client.
- ✓ Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- ✓ Thorough line-by-line manual review of the entire codebase by industry experts.

Blockchain security tools used:

- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat



# TOKEN TRANSFERS STATS

Transfer Count	TESTNET
Uniq Senders	TESTNET
Uniq Receivers	TESTNET
Total Amount	TESTNET
Median Transfer Amount	TESTNET
Average Transfer Amount	TESTNET
First transfer date	TESTNET
Last transfer date	TESTNET
Days token transferred	TESTNET

# SMART CONTRACT STATS

Calls Count	TESTNET
External calls	TESTNET
Internal calls	TESTNET
Transactions count	TESTNET
Uniq Callers	TESTNET
Days contract called	TESTNET
Last transaction time	TESTNET
Created	TESTNET
Create TX	TESTNET
Creator	TESTNET



# FEATURED WALLETS

Owner address	0xBA799d418D1356ff5d225096d08951a3b45b6e4A
Marketing fee receiver	N/A
LP address	N/A

# TOP 3 UNLOCKED WALLETS

N/A	N/A
N/A	N/A
N/A	N/A



# VULNERABILITY ANALYSIS

ID	Title	
SWC-100	Function Default Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Passed
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed
SWC-107	Reentrancy	Passed
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Passed
SWC-116	Block values as a proxy for time	Passed
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed



# VULNERABILITY ANALYSIS

ID	Title	
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Griefing	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed
SWC-134	Message call with hardcoded gas amount	Passed
SWC-135	Code With No Effects	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed





# FOUND THREATS

## High Risk

No high risk-level threats found in this contract.

## Medium Risk

No medium risk-level threats found in this contract.



# FOUND THREATS

## ⚠ Low Risk

**Extreme caution should be taken when deploying new contract versions.**  
When deployed, the contract by default is not initialized and without owner.  
The initialize function sets msg.sender as owner and can be called by anyone.

```
bool internal _initialized;

function initialize() external {
    require(!_initialized);

    _transferOwnership(msg.sender);
    _initialized = true;
}

function _transferOwnership(address newOwner) internal {
    address oldOwner = _owner;
    _owner = newOwner;

    emit OwnershipTransferred(oldOwner, newOwner);
}

modifier onlyOwner() virtual override {
    require(msg.sender == _owner ||
        (MULTISIGN_ADDRESS != address(0) && msg.sender == MULTISIGN_ADDRESS), "Unauthorized");
    _;
}
```

- Recommendation:
  - Always initialize the new implementation of Factory contract when assigning it to the main proxy contract to prevent hijacking.



# FOUND THREATS

## Informational

**Owner can add credits to already existing users.**

```
function addCredit(address user, uint256 amount) external onlyOwner nonReentrant {
    require(amount > 0);
    require(_userData[user].exists, "Unknown user");

    unchecked {
        _userData[user].balance += amount;
        _userData[user].addedCredit.push(addedCreditData(_timestamp(), amount, msg.sender));
        emit AddedCredit(user, amount);
    }
}
```

**Owner can set/change discount levels.**

Users get discounts on contract's deployment price by holding or staking \$FACTORY tokens.

```
function setDiscountLevel(uint8 level, uint24 percent, uint24 discount) external onlyOwner {
    require(level < 3);

    unchecked {
        require(percent <= denominator * 100);

        if (level + 1 < 3 && _discountLevel[level + 1].exists) { require(_discountLevel[level + 1].percent > percent); }
        if (level - 1 >= 0 && _discountLevel[level - 1].exists) { require(_discountLevel[level - 1].percent < percent); }
    }

    if (!_discountLevel[level].exists) { _discountLevel[level].exists = true; }

    _discountLevel[level].percent = percent;
    _discountLevel[level].discount = discount;

    emit SetDiscountLevel(level, percent, discount);
}
```



# FOUND THREATS

## Informational

**Owner can change multi sig and treasury addresses.**

```
function setMultiSignatureWallet(address account) external onlyOwner {
    MULTISIGN_ADDRESS = account;
}

function setTreasury(address payable account) external onlyOwner {
    TREASURY_ADDRESS = account;
}
```

**Owner can change factory and stake tokens.**

```
function setFactoryInterfaces(address token, address stake) external onlyOwner {
    FACTORY_TOKEN = IERC20(token);
    FACTORY_STAKE = IStake(stake);
}
```

**Owner can change factory and stake tokens.**

```
function recoverETH(address payable to, uint256 amount) external onlyOwner {
    (bool success, ) = to.call{ value: amount }("");
    require(success);
}

function recoverERC20(address token, address to, uint256 amount) external onlyOwner {
    IERC20(token).transfer(to, amount);
}
```



# FOUND THREATS

## Informational

**Owner can set new templates and change existing ones.**

```
function setTemplate(uint256 templateId, bool active,
bool discountable, uint256[] calldata price) external onlyOwner {
    uint256 cnt = price.length;

    require(cnt > 0 && cnt < 64);

    if (!_templateData[templateId].exists) {
        _templateData[templateId].exists = true;
        _templateList.push(templateId);
    }

    _templateData[templateId].active = active;
    _templateData[templateId].discountable = discountable;

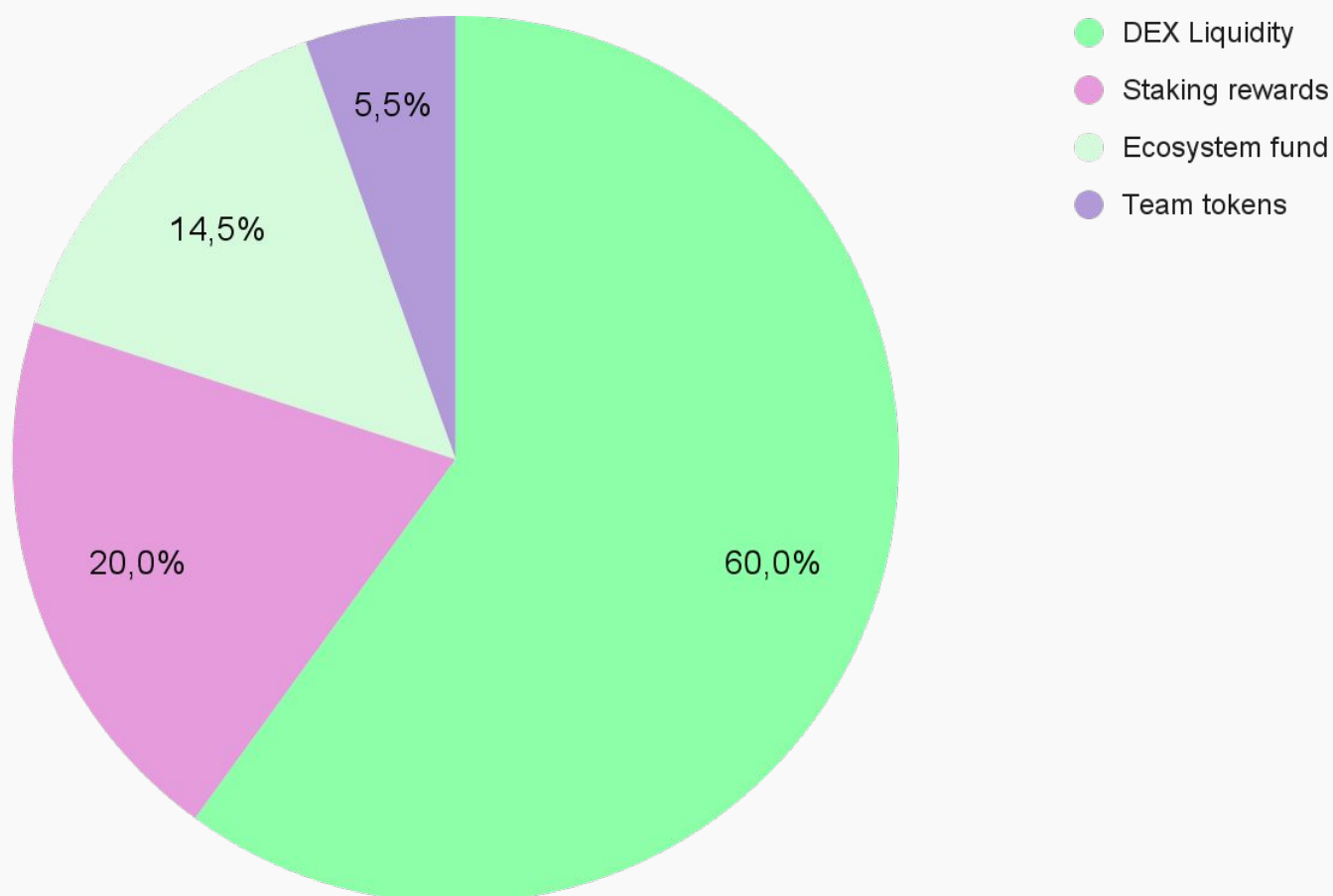
    unchecked {
        _templateData[templateId].features = cnt - 1;

        for (uint256 f; f < cnt; f++) { _templateData[templateId].price[f] = price[f]; }
    }
}
```



The following tokenomics are based on the project's whitepaper and/or website:

- 60% - DEX Liquidity
- 14.5% - Ecosystem fund
- 20% - Staking rewards
- 5.5% - Team tokens



TOKENOMICS



# WEBSITE

## Website URL

<https://chainfactory.app/>

## Domain Registry

<https://domains.google.com>

## Domain Expiration

2024-05-28

## Technical SEO Test

Passed

## Security Test

Passed. SSL certificate present

## Design

Simple and intuitive web design with appropriate color scheme and graphics.

## Content

The information helps new investors understand what the product does right away.

No grammar mistakes found.

## Whitepaper

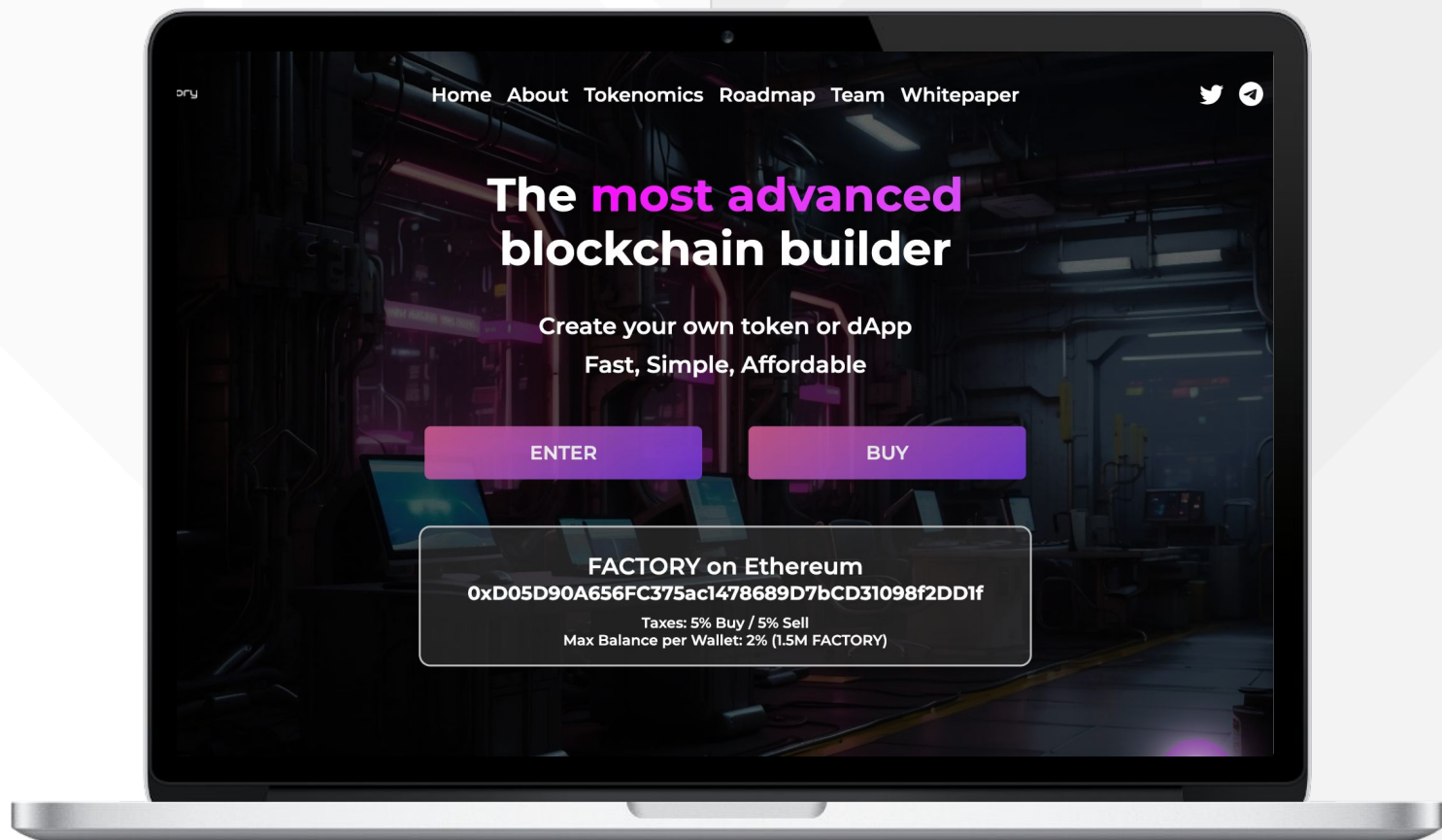
Well written, explanatory.

## Roadmap

Yes, goals set without time frames.

## Mobile-friendly?

Yes



# chainfactory.app





# SOCIAL MEDIA & ONLINE PRESENCE



## ANALYSIS

Project's social media pages are active.



### Twitter's X

@ChainFactoryApp

- 157 followers
- Posts frequently
- Active



### Discord

<https://discord.com/invite/4eDJf6UwP4>

- 83 members
- Active members
- Active mods



### Telegram

@ChainFactoryVerify

- 404 members
- Active members
- Active mods



### Medium

- Not available





# SPYWOLF

## CRYPTO SECURITY

Audits | KYCs | dApps  
Contract Development

## ABOUT US

We are a growing crypto security agency offering audits, KYCs and consulting services for some of the top names in the crypto industry.

- ✓ OVER 700 SUCCESSFUL CLIENTS
- ✓ MORE THAN 1000 SCAMS EXPOSED
- ✓ MILLIONS SAVED IN POTENTIAL FRAUD
- ✓ PARTNERSHIPS WITH TOP LAUNCHPADS, INFLUENCERS AND CRYPTO PROJECTS
- ✓ CONSTANTLY BUILDING TOOLS TO HELP INVESTORS DO BETTER RESEARCH

To hire us, reach out to  
[contact@spywolf.co](mailto:contact@spywolf.co) or  
[t.me/joe\\_SpyWolf](https://t.me/joe_SpyWolf)

## FIND US ONLINE



[SPYWOLF.CO](https://spywolf.co)



[@SPYWOLFNETWORK](https://t.me/SPYWOLFNETWORK)



[@SPYWOLFNETWORK](https://t.me/SPYWOLFNETWORK)



# Disclaimer

This report shows findings based on our limited project analysis, following good industry practice from the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, overall social media and website presence and team transparency details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report.

While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

## **DISCLAIMER:**

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice.

No one shall have any right to rely on the report or its contents, and SpyWolf and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (SpyWolf) owe no duty of care towards you or any other person, nor does SpyWolf make any warranty or representation to any person on the accuracy or completeness of the report.

The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SpyWolf hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SpyWolf hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SpyWolf, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts, website, social media and team.

No applications were reviewed for security. No product code has been reviewed.

